Theoretical Computer Science 31 (1984) 125–138
North-Holland

NEW NP-HARD AND NP-COMPLETE POLYNOMIAL AND INTEGER DIVISIBILITY PROBLEMS

David A. PLAISTED*
Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, U.S.A.

Communicated by M.S. Paterson
Received March 1978
Revised October 1983

Abstract. We show that some problems involving sparse polynomials are NP-hard. For example, it is NP-hard to determine if a sparse polynomial has a root of modulus 1, and it is NP-hard to decide if two sparse polynomials are not relatively prime. Also, we show that a divisibility problem involving an unbounded number of sparse polynomials is NP-complete using a theorem of Linnik concerning the distribution of primes in arithmetic sequences. From these results it follows that certain problems involving inequalities, recurrence relations, differential equations, and eigenvalues of sparse matrices are NP-hard. Problems involving divisibility properties of two sparse polynomials, divisibility of sparse binary numbers, and ring homomorphisms are also NP-hard.

1. Introduction

The computational complexity of some problems involving divisibility of sparse polynomials and integers has been studied in [12, 13]. We extend these results in several ways. Using a theorem from Linnik [10] on the distribution of primes in arithmetic sequences, we show a problem to be NP-complete which was only known to be NP-hard previously. Also, we exhibit some NP-hard problems involving divisibility of one or two sparse polynomials. (The problems in [12, 13] all involved an unbounded number of sparse polynomials.) For example, it is NP-hard to determine if two sparse polynomials have a nontrivial greatest common divisor. These results are applied to obtain NP-hard problems involving inequalities, recurrence relations, differential equations, and eigenvalues of sparse matrices. Although it is still unknown how hard it is to determine whether one sparse polynomial divides another, we give two closely related problems that are NP-hard. Also, a problem involving sparse binary numbers and a problem involving ring homomorphisms are discussed. The analysis of the above problems makes use of polynomials that are generalizations of those in [13]; we discuss some properties of these polynomials and give algorithms for computing them. The results presented here are of interest

* Present address: 222 Digital Computer Laboratory, 1304 West Springfield Avenue, Urbana, IL 61801, U.S.A.

because of the importance of polynomial divisibility properties in algebraic coding theory, and also because of their implications for symbolic manipulation of algebraic expressions. The theoretical value of these results is that some of the NP-hard problems have the nondeterminism 'hidden'. That is, the problems are not explicitly stated in terms of one of a large number of possibilities being true. Few other known NP-hard problems have this property. For results concerning the intractability of problems in number theory, see [1, 2, 7, 11].

In the following discussion, we consider a polynomial $\sum_j a_j x^j$ to be represented as a sequence of ordered pairs (a_j, j) corresponding to non-zero coefficients a_j. A polynomial represented in this way will be termed a sparse polynomial. Note that it is the representation of these problems that makes them difficult rather than the sparseness of the polynomials. The representation allows for polynomials of high degree to be written down in a short space. From now on, polynomials will be assumed to have integer coefficients unless otherwise specified.

2. Definitions

The following conventions will be used: P is the class of sets S of strings such that there exists a deterministic Turing machine M and a polynomial p such that M accepts S and for all inputs x to M, M halts within $p(|x|)$ steps (where $|x|$ is the length of x in characters). NP is the class of sets S of strings such that there exists a nondeterministic Turing machine M and a polynomial p such that M accepts S and for all inputs x to M, all computation paths of M halt within $p(|x|)$ steps. Also, CoNP is the class of sets S such that the complement of S is in NP. It is widely believed that $P \neq NP \neq CoNP \neq P$; for a discussion of these questions and their importance, see [4]. A set S is said to be *tractable* if S is in P; otherwise S is *intractable*. We say a function f mapping strings to strings is computable in polynomial time if there is a Turing machine M and a polynomial p such that for all elements x in the domain of f, M outputs $f(x)$ when given input x, and halts within $p(|x|)$ steps. If A and B are sets of strings, we write $A \leq_p B$ (A is polynomial reducible to B) if there is a function f computable in polynomial time such that for all strings x, $x \in A$ iff $f(x) \in B$. Such a function f is called a polynomial time reduction of A to B. A set B is *NP-hard* if for all A in NP, $A \leq_p B$. A set B is *NP-complete* if it is in NP and is NP-hard. Many such NP-complete sets are known. Note that if B is NP-complete and $B \in P$, then $P = NP$; this is considered good evidence that B is not in P. Also, if any NP-hard set is in CoNP then NP = CoNP. A set S is *CoNP-hard* if the complement of S is NP-hard. For convenience we say that a *function* g is NP-hard if $\{(x, y): y = g(x)\}$ is NP-hard. We call $\{(x, y): y = g(x)\}$ the *graph* of g. Note that g is NP-hard iff the graph of g (as a set) is NP-hard. Similarly, we say g is CoNP-hard if the complement of the graph of g is NP-hard. By the complement of the graph of g we mean the set $\{(x, y): y \neq g(x)\}$. If g is NP-hard, this is evidence that the graph of g is not in P; if g is CoNP-hard, this is evidence that the graph

of g is not in NP. We will introduce several natural functions which are CoNP-hard, and thereby obtain several natural sets that are CoNP-hard. From now on we identify a set S with the problem of determining whether a string x belongs to S. Thus we can speak of a problem as being NP-hard or NP-complete. Also, we assume integers are encoded in binary so that integers and sparse polynomials can be represented as character strings in some standard way. The letters x, y, and z will usually represent real or complex variables in the following discussion.

3. Polynomials used to obtain the reductions

We introduce polynomials which are used to obtain many of the results concerning NP-hardness of polynomial and integer divisibility problems. These polynomials are slight generalizations of those in [13]; the method used to define them here is new. Also, we present some identities which are useful for computing and manipulating these polynomials.

Let q_j be the jth prime number. Let M be an integer and let W be a well-formed formula of the propositional calculus obtained from predicate symbols P_j using Boolean connectives (including negation). With each Mth root of unity ω in the complex plane (i.e., a solution of the equation $z^M = 1$), we associate an *interpretation* $I_M(\omega)$ of the predicate symbols $\{P_j\colon q_j \text{ divides } M\}$. In particular, the interpretation $I_M(\omega)$ makes predicate symbol P_j true iff $\omega^{M/q_j} = 1$. It is not difficult to show that for all interpretations J of $\{P_j\colon q_j \text{ divides } M\}$, there exists at least one Mth root of unity ω such that $I_M(\omega) = J$. This is important.

Theorem 3.1. *Suppose M is an integer and W is a well-formed formula of the propositional calculus. Suppose that all predicate symbols occurring in W are of the form P_j for some j such that q_j divides M. Then there is a unique polynomial having the following properties:*

(1) *$p(z) = 0$ iff z is an M-th root of unity and the wff W is true in interpretation $I_M(z)$.*

(2) *$p(z)$ has no zeroes of multiplicity greater than one.*

(3) *The leading coefficient of $p(z)$ is 1.*

Proof. The zeroes of p are uniquely specified by (1) and (2). This determines p up to a constant factor. This factor is determined by (3). □

Definition. With notation as above, $\text{Poly}_M(W)$ is the unique polynomial having properties (1), (2) and (3) above.

Definition. A *literal* is a formula of the form P or $\neg P$ where P is a predicate symbol. A *clause* is a disjunction of literals.

Theorem 3.2. *The polynomials* $\text{Poly}_M(W)$ *have the following properties:*

(1) $\mathrm{Poly}_M(A)$ *has integer coefficients for all* A.
(2) $\mathrm{Poly}_M(A) = z^M - 1$ *iff* A *is valid.*
(3) $\mathrm{Poly}_M(A) = 1$ *iff* A *is inconsistent.*
(4) $\mathrm{Poly}_M(A1) = \mathrm{Poly}_M(A2)$ *iff* $A1 \equiv A2$.
(5) $\mathrm{Poly}_M(P_j) = x^{M/q_j} - 1$.
(6) $\mathrm{Poly}_M(\neg A) = (x^m - 1)/\mathrm{Poly}_M(A)$.
(7) $\mathrm{Poly}_M(A \wedge B) = \gcd(\mathrm{Poly}_M(A), \mathrm{Poly}_M(B))$.
(8) $\mathrm{Poly}_M(A \vee B) = \mathrm{lcm}(\mathrm{Poly}_M(A), \mathrm{Poly}_M(B))$.
(9) *If* C *is a 3-literal clause and* M *has only small prime factors then* $\mathrm{Poly}_M(C)$ *and* $\mathrm{Poly}_M(\neg C)$ *can be computed as sparse polynomials in polynomial time (and are of polynomial length as character strings).*

Proof. For (1), note that $\mathrm{Poly}_M(W)$ is a product of cyclotomic polynomials which are known [8] to have integer coefficients. Properties (2) through (8) are easy. Property (9) will be shown using identities given below. □

3.1. Some identities

Given a wff W, let $W\{P_j \leftarrow \text{TRUE}\}$ be W with the predicate symbol P_j replaced by TRUE everywhere. Similarly define $W\{P_j \leftarrow \text{FALSE}\}$. The following identities can be derived without much difficulty:

(I) $\mathrm{Poly}_M(P_j \wedge W) = \mathrm{Poly}_{M/q_j}(W\{P_j \leftarrow \text{TRUE}\})$.
(II) $\mathrm{Poly}_M(\bar{P}_j \vee W) = ((x^M - 1)/(x^{M/q_j} - 1))\,\mathrm{Poly}_{M/q_j}(W\{P_j \leftarrow \text{TRUE}\})$.
(III) If P_j does not occur in W and q_j divides M, then $\mathrm{Poly}_M(W)(x) = \mathrm{Poly}_{M/q_j}(W)(x^{q_j})$.
(IV) If q_j^2 divides M then $\mathrm{Poly}_M(W)(x) = \mathrm{Poly}_{M/q_j}(W)(x^{q_j})$ regardless of whether P_j occurs in W.
(V) $\mathrm{Poly}_M(W1 \vee W2) = \mathrm{Poly}_M(W1)\,\mathrm{Poly}_M(W2)/\mathrm{Poly}_M(W1 \wedge W2)$.
(VI) $\mathrm{Poly}_M(P_{r_1} \wedge \cdots \wedge P_{r_t}) = x^{M/(r_1 r_2 \cdots r_t)} - 1$ if the r_i are distinct primes.

These identities are useful in showing that if C is a 3-literal clause and M has only small prime factors, then $\mathrm{Poly}_M(C)$ can be computed in polynomial time. Using (III) and (IV) we can reduce the evaluation of $\mathrm{Poly}_M(C)$ to the evaluation of $\mathrm{Poly}_{M1}(C)$ where $M1$ is the product of 3 distinct small primes. Negative literals can be eliminated from C using (II). This costs one polynomial multiplication, but that can be done in polynomial time since $M1$ is polynomial in $\log(M)$. Suppose C contains 3 positive literals; thus $C = P_a \vee P_b \vee P_c$ and $M1 = q_a q_b q_c$. In this case one can show that

$$\mathrm{Poly}_{M1}(C) = \frac{(x^{q_a q_b} - 1)(x^{q_b q_c} - 1)(x^{q_a q_c} - 1)(x - 1)}{(x^{q_a} - 1)(x^{q_b} - 1)(x^{q_c} - 1)}.$$

This quotient can be computed in polynomial time using the standard dense polynomial multiplication and division algorothms since q_a, q_b and q_c are small. If C is $\{P_a \vee P_b\}$ or if C is $\{P_a\}$, the expression for $\mathrm{Poly}_{M1}(C)$ is even simpler. In all cases $\mathrm{Poly}_{M1}(C)$ can be computed in polynomial time. Also, $\mathrm{Poly}_M(\neg C)$ can be computed

from $\text{Poly}_M(C)$ in polynomial time using property (6) of Theorem 3.2. Hence $\text{Poly}_M(\neg C)$ can also be computed in polynomial time if M has only small prime factors since the evaluation of $\text{Poly}_M(\neg C)$ can be reduced to the evaluation of $\text{Poly}_{M1}(\neg C)$ as above for some $M1$ which is the product of 3 small primes.

The identities (I) through (VI) are also useful in showing that if C is a 3-literal clause having at most 2 positive literals, or C is a 2-literal clause, then $\text{Poly}_M(C)$ and $\text{Poly}_M(\neg C)$ have all coefficients in the set $\{-1, 0, 1\}$. This is slightly more general than the result given earlier [13], since M is not necessarily the product of the first n primes for some n, and since M is not necessarily square-free.

In particular, using (III) and (IV) we can reduce the evaluation of $\text{Poly}_M(C)$ and $\text{Poly}_M(\neg C)$ to the case where M is square-free and the product of 2 or 3 distinct prime factors, if C is a 2-literal clause or a 3-literal clause. Using (I), we can eliminate negative literals of C from consideration when evaluating $\text{Poly}_M(\neg C)$. Using (II), it is not difficult to show that we can eliminate negative literals of C from consideration when evaluating $\text{Poly}_M(C)$ without affecting the question of whether all coefficients are $+1$, 0 or -1. Finally, we can show that if C is an all-positive clause having 2 or fewer literals, then $\text{Poly}_M(C)$ and $\text{Poly}_M(\neg C)$ only have coefficients of 0, 1 and -1. The only non-trivial cases are $\text{Poly}_{q_i q_j}(P_i \vee P_j)$ and $\text{Poly}_{q_i q_j}(\bar{P}_i \wedge \bar{P}_j)$, which can easily be evaluated using (V) and (VI) and shown to have coefficients of 1, 0 and -1 only.

3.2. A sample application

We review a result of [13] to show how easy it is to obtain NP-hardness results using the polynomials $\text{Poly}_M(W)$.

Theorem 3.3. *The following problem is* NP*-hard: Given a set* $\{p_1(x), \ldots, p_k(x)\}$ *of sparse polynomials with integer coefficients, to determine if they have a nontrivial greatest common divisor.*

Proof. We reduce from 3-satisfiability. Suppose $S = \{C1, \ldots, Ck\}$ is a set of 3-literal clauses over the predicate symbols $\{P1, \ldots, Pn\}$. Let M be $q_1 q_2 \cdots q_n$, the product of the first n primes. Then S is satisfiable *iff* there exists an interpretation I of $\{P1, \ldots, Pn\}$ making all C in S true, iff there exists an Mth root of unity ω such that $I_M(\omega) = I$ for some interpretation I making all clauses C in S true, iff there exists an Mth root of unity ω such that $\text{Poly}_M(C_j)(\omega) = 0$ for $j = 1, \ldots, k$, *iff* $\gcd(\text{Poly}_M(C1), \ldots, \text{Poly}_M(Ck))$ has degree greater than zero. □

4. Problems in NP

We now show a problem to be NP-complete that was only known to be NP-hard previously. The proof makes use of a theorem of Linnik on the distribution of primes in arithmetic sequences. Consider the following problem.

SPARSE-POLY-DIVIS. Given an integer N and a finite set $\{p_1(x), \ldots, p_k(x)\}$ of sparse polynomials. Then the problem is to determine whether $x^N - 1$ is *not* a factor of $\prod_{j=1}^{k} p_j(x)$.

Theorem 4.1. SPARSE-POLY-DIVIS *is* NP-*complete.*

Proof. We showed in [13] that this problem is NP-hard. This proof can be done elegantly as above using the polynomials $\text{Poly}_M(W)$. We now show that it is in NP. The method is as follows: The polynomial $x^N - 1$ is a factor of $\prod_{j=1}^{k} p_j(x)$ iff the polynomial $x^{cN} - 1$ is a factor of $\prod_{j=1}^{k} p_j(x^c)$, for any integer $c > 0$. Suppose $cN + 1$ is a prime number (call it q). Then, by Fermat's theorem, $b^{cN} - 1 \equiv 0 \pmod q$ for all integers b that are not multiples of q. Hence $\prod_j p_j(b^c) \equiv 0 \pmod q$ for all such b if $x^N - 1$ is a factor of $\prod_j p_j(x)$. We desire to show that if c is suitably chosen, then $x^N - 1$ is a factor of $\prod_{j=1}^{k} p_j(x)$ *iff* $\prod_j p_j(b^c) \equiv 0 \pmod q$ for all b that are not multiples of q. If c is not too large, this latter condition can be tested nondeterministically, giving the desired result that SPARSE-POLY-DIVIS is in NP.

Let $r(x)$ be the remainder when $\prod_j p_j(x)$ is divided by $x^N - 1$. If $\prod_j p_j(b^c) \equiv 0 \pmod q$ for all integers b that are not multiples of q, then $r(b^c) \equiv 0 \pmod q$ for all integers b that are not multiples of q. Since there are $q - 1$ congruence classes of such integers, and since the degree of $r(x^c)$ is less than $q - 1$ for $c > 0$, it follows by elementary number theory that all coefficients of $r(x^c)$ and hence of $r(x)$ must be multiples of q. If q is chosen large enough, this cannot happen unless all coefficients of $r(x)$ are *zero*, and thus $x^N - 1$ divides $\prod_{j=1}^{k} p_j(x)$. Hence we can nondeterministically show that $x^N - 1$ does not divide $\prod_j p_j(x)$ by finding appropriate integers c and b such that $\prod_{j=1}^{k} p_j(b^c) \not\equiv 0 \pmod{cN+1}$.

In particular, given a polynomial p, let $\|p\|$ be the sum of the absolute values of the coefficients of p. It is not difficult to show that

$$\|r(x)\| \leq \prod_{j=1}^{k} \|p_j(x)\|.$$

This inequality was derived in [13]. Also, the largest coefficient in $r(x)$ can certainly never be larger than $\|r(x)\|$ in absolute value. Hence it suffices to choose c so that q is larger than $\prod_{j=1}^{k} \|p_j(x)\|$. Note that $\log \prod_{j=1}^{k} \|p_j(x)\|$ is polynomial in the length of the input.

We now consider the question of the existence of primes having the desired properties. It is known [9] that primes exist in all arithmetic progressions, and so we know that a large enough prime q of the form $cN + 1$ exists. The problem is that if q is too large, then we cannot compute $(\prod_j p_j(x)) \pmod q$ in polynomial time for an arbitrary integer x in the range $0 \leq x \leq q - 1$. We use a result of Linnik [10] to show that there is always a prime of the required form that is not too large.

Theorem 4.2 (Linnik [10]). *The least prime in the arithmetic progression $Dx + \lambda$, where D and λ are relatively prime and $1 \leq \lambda \leq D - 1$, does not exceed D^{c_0}. Here c_0 is an absolute constant.*

If one assumes the extended Riemann hypothesis, one can show [6] that c_0 is near 2.

Proof of Theorem 4.1 (*continued*). Let D be the least multiple of N that is greater than or equal to $\prod_j \|p_j(x)\|$. Then if $N \leq \prod_j \|p_j(x)\|$, $D \leq 2\prod_j \|p_j(x)\|$ and if $N \geq \prod_j \|p_j(x)\|$, $D \leq N$. By Linnik's theorem, there is a prime q of the form $Dx+1$ with $q \leq \max(N, 2\prod_j \|p_j(x)\|)^{c_0}$. Now $q \neq 1$ since 1 is not prime. Hence $D+1 \leq q \leq D^{c_0}$. Also, since $D \geq \prod_j \|p_j(x)\|$, $q > \prod_j \|p_j(x)\|$ as desired. □

In summary, we can solve the problem SPARSE-POLY-DIVIS nondeterministically in polynomial time, as follows:

Given N and $\{p_1(x), \ldots, p_k(x)\}$, we want to determine if $x^N - 1$ is not a factor of $\prod_j p_j(x)$. To do this, we nondeterministically choose c, $c > 0$, such that

(a) $cN+1$ is prime,

(b) $cN+1 > \prod_{j=1}^{k} \|p_j\|$,

(c) $cN+1$ is not larger than the bound given by Linnik's theorem.

We can verify that $cN+1$ is prime in nondeterministic polynomial time since short certificates of primality always exist [14]. We can also nondeterministically choose b such that $\prod_j p_j(b^c) \neq 0 \pmod{cN+1}$ if such an integer b exists. Furthermore, we can evaluate $\prod_j p_j(b^c) \pmod{cN+1}$ in polynomial time since $cN+1$ is not too big. If such b and c can be found, then $x^N - 1$ is *not* a factor of $\prod_j p_j(x)$, otherwise it is. In fact, SPARSE-POLY-DIVIS is still NP-complete if we restrict the non-zero coefficients of the polynomials p_j to be ± 1.

4.1. Evaluating expressions

The methods used to show that SPARSE-POLY-DIVIS is in NP can also be used to show that some problems involving inequations of algebraic expressions are in NP. Consider the following problem.

SPARSE-POLY-NONROOT. Given a sparse polynomial p with integer coefficients and an integer M. Then the problem is to determine if $p(\omega) \neq 0$ where ω is a primitive Mth root of unity.

Theorem 4.3. SPARSE-POLY-NONROOT *is in* NP.

Proof. Suppose ω is a primitive Mth root of unity. Then $p(\omega) = 0$ iff $z^M - 1$ divides $p(z) \prod \{z^{M/q} - 1 : q \text{ prime}, q \text{ divides } M\}$. This is because all roots of $z^M - 1$ that are not primitive Mth roots of unity will be roots of some $z^{M/q} - 1$, and if *any* primitive Mth root of unity is a root of p, then *all* primitive Mth roots of unity are. But we can determine that $z^M - 1$ is *not* a factor of a product of polynomials in nondeterministic polynomial time, as shown above. Also, we can factor M in nondeterministic polynomial time if necessary, using Pratt's result that short certificates of primality

exist [14]. Therefore SPARSE-POLY-NONROOT is in NP. The author believes he has a method for solving problem SPARSE-POLY-NONROOT in polynomial time if the prime factorization of M is given. □

5. Problems involving one or two sparse polynomials

We now exhibit some NP-hard problems involving one or two sparse polynomials. Also, we give some easy applications of these results. These problems contrast with those of [13] which all involve an unbounded number of sparse polynomials. In particular, we show that the following problems are all NP-hard.

SPARSE-POLY-ROOT-MODULUS-1. Given a sparse polynomial $p(z)$ with integer coefficients. Then the problem is to determine if p has a root r of modulus 1 (that is, a root on the complex unit circle).

2-SPARSE-POLY-GCD. Given two sparse polynomials $p_1(z)$ and $p_2(z)$ with integer coefficients. Then the problem is to determine whether $p_1(z)$ and $p_2(z)$ are not relatively prime (that is, the degree of $\gcd(p_1(z), p_2(z))$ is greater than zero).

TRIG-INEQUALITY. Given integers $a_1, a_2, \ldots, a_n, b_1, b_2, \ldots, b_n$ and c. Then the problem is to determine whether there exists a real θ such that the inequality

$$c + \sum_{i=1}^{n} a_i \cos(b_i \theta) > 0$$

fails to hold.

BOUNDED RECURRENCE. Given a recurrence relation, the problem is to determine if it has a bounded (i.e., periodic) integer solution.

For this problem, a recurrence relation

$$\alpha_n = \sum_{i=1}^{m} c_i \alpha_{n-b_i}$$

is represented as a sequence $(c_1, b_1), \ldots, (c_m, b_m)$ of ordered pairs, where only non-zero c_i are included.

SPARSE MATRIX EIGENVALUE. Given a sparse $N \times N$ integer matrix A, the problem is to determine if $J_N + A$ has an eigenvalue of modulus *exactly* one. Here J_N is the $N \times N$ matrix such that $J_N(i, j) = 1$ if $j = i + 1$, $J_N(i, j) = 0$ otherwise.

COMMON-DIFF-EQ-SOLUTION. Given two sparse homogeneous linear differential equations in one variable with constant coefficients, the problem is to determine if they have a common nontrivial solution.

From the fact that the above problems are all NP-hard, it follows that the following functions are CoNP-hard: The number of roots of modulus 1 of a sparse polynomial with integer coefficients, the degree of the greatest common divisor of two sparse polynomials with integer coefficients, the degree of the last common multiple of two such sparse polynomials, and the number of values θ making the expression of TRIG-INEQUALITY equal to zero. Recall that if f is CoNP-hard, the graph of f is CoNP-hard as a set. In this way, we obtain natural CoNP-hard sets from these CoNP-hard functions.

Theorem 5.1. SPARSE-POLY-ROOT-MODULUS-1, 2-SPARSE-POLY-GCD, *and* TRIG-INEQUALITY *are* NP-*hard.*

Proof. Given a complex number $z = x + iy$, let $\mathrm{Conj}(z)$ denote $x - iy$, the complex conjugate of z. It is easy to verify that $\mathrm{Conj}(z1 z2) = \mathrm{Conj}(z1)\,\mathrm{Conj}(z2)$, $\mathrm{Conj}(z1 + z2) = \mathrm{Conj}(z1) + \mathrm{Conj}(z2)$, and if $z = x + iy$ then $z\,\mathrm{Conj}(z) = x^2 + y^2$, a real, non-negative quantity. Also, if $p(z)$ is a polynomial with real coefficients, then $p(z) = 0$ iff $p(\mathrm{Conj}(z)) = 0$. Furthermore, if z is on the complex unit circle, then $\mathrm{Conj}(z) = 1/z$.

From these facts, it follows that if $p(z)$ is a polynomial with real coefficients, then $p(z)p(1/z)$ is real and non-negative on the complex unit circle, and has zeroes on the complex unit circle exactly where $p(z)$ does. Therefore, if S is a set of clauses, and M is chosen properly, then $\Sigma\{\mathrm{Poly}_M(C)(z)\,\mathrm{Poly}_M(C)(1/z)\colon C \in S\}$ has a zero on the complex unit circle (in fact, at an Mth root of unity) iff the $\mathrm{Poly}_M(C)$ have a common root, i.e., iff S is consistent. Define $t_M(S)$ to be $\Sigma\{\mathrm{Poly}_M(C)(z)\,\mathrm{Poly}_M(C)(1/z)\colon C \in S\}$. Then $z^M t_M(S)$ is a sparse polynomial with integer coefficients, and has a zero on the complex unit circle (at an Mth root of unity) iff S is consistent. Furthermore, if S is a set of 3-literal clauses, then the coefficients of the sparse polynomial $z^M t_M(S)$ can be computed in polynomial time (assuming that M has only small prime factors). Thus SPARSE-POLY-ROOT-MODULUS-1 is NP-hard. Also, 2-SPARSE-POLY-GCD is NP-hard because S is consistent iff the degree of the greatest common divisor of $z^M - 1$ and $z^M t_M(S)$ is greater than zero. Note that it is also NP-hard to determine if the degree of the least common multiple of two sparse polynomials is not equal to a given integer.

To show that TRIG-INEQUALITY is NP-hard, we proceed as follows: Since $t_M(S)(z) = t_M(S)(1/z)$, the function $t_M(S)$ can easily be expressed as a constant plus a sum of terms of the form $a(z^b + z^{-b})$ for various integers a and b. For z on the complex unit circle, we have that $z = e^{i\theta}$ and so such a term is equal to $2a\cos(b\theta)$. Furthermore, $t_M(S)$ is everywhere positive on the complex unit circle iff S is inconsistent. Letting c be the constant term in $t_M(S)$, we easily obtain that TRIG-INEQUALITY is NP-hard. □

Note that a root u as in SPARSE-POLY-ROOT-MODULUS-1 need not be a root of unity. Even if p has leading coefficient 1 and integer coefficients, it is possible that

$p(z)=0$ and $|z|=1$ but z is not a root of unity. (If all such roots were roots of unity, one might try to solve SPARSE-POLY-ROOT-MODULUS-1 by guessing such a root.) For example, let u be $(1-\xi r)/(1-\xi^2 r)$, where $r=2^{1/3}$ and $\xi=e^{2\pi i/3}$. Then $|u|=1$ but one can show that u is not a root of unity. Also, u is an algebraic integer, that is, a root of a polynomial with integer coefficients and leading coefficient 1. This follows because r and ξ are clearly algebraic integers ($r^3-2=0$, $\xi^3-1=0$) and u can be expressed as $(r^2+r-3)+(-r^2+2r-2)\xi$. It is known [3] that sums and products of algebraic integers are algebraic integers, so u is an algebraic integer. This example was given by S. Ullom.

Corollary 5.2. BOUNDED-RECURRENCE *is* NP-*hard.*

Proof. The function $(z^M-1)(z^{-M}-1)$ is real and nonnegative on the complex unit circle, with zeroes at all the Mth roots of unity but nowhere else. Also, $t_M(S)(x)$ is real and nonnegative on the complex unit circle, and all its zeroes on the unit circle are at Mth roots of unity. Hence $(z^M-1)(z^{-M}-1)+t_M(S)(z)$ has a zero on the unit circle iff $t_M(S)$ does, and if so it is at an Mth root of unity. Let $p(z)$ be $z^M[(z^M-1)(z^{-M}-1)+t_M(S)(z)]$. Note that $p(z)$ is easy to compute as a sparse polynomial from S. Also, $p(z)$ has a zero on the unit circle iff $t_M(S)$ does, and if so it is at an Mth root of unity. But $p(z)$ has integer coefficients and has a leading coefficient of 1. Hence the recurrence relation corresponding to $p(z)$ has a periodic integer solution iff $p(z)$ has a root on the complex unit circle, i.e., iff S is consistent. Thus BOUNDED RECURRENCE is NP-hard. (The recurrence relation corresponding to the polynomial

$$z^n-\sum_{i=1}^{M} c_i z^{n-b_i}$$

is $(c_1, b_1), \ldots, (c_m, b_m)$.) □

Corollary 5.3. SPARSE MATRIX EIGENVALUE *and* COMMON-DIFF-EQ-SOLUTION *are also* NP-*hard.*

Proof. For SPARSE MATRIX EIGENVALUE, we can easily construct a matrix A such that $J_N + A$ has an eigenvalue of modulus one iff a given sparse recurrence relation has a periodic integer solution. For COMMON-DIFF-EQ-SOLUTION, with the differential equation

$$\sum_i a_i \frac{d^i x}{dt^i}=0,$$

we associate the polynomial $\sum_i a_i x^i$. It turns out that the two differential equations have a common nontrivial solution iff their polynomials have a common root.

6. Divisibility of two sparse polynomials

The following problems relate to divisibility properties of two sparse polynomials It is not known whether the problem of deciding if one sparse polynomial divides another is in P. Nor is this problem known to be NP-complete.

Consider the following problems.

SPARSE-POLY-QUOTIENT. Given two sparse polynomials $p_1(x)$ and $p_2(x)$ with integer coefficients, the problem is to determine if the quotient when $p_1(x)$ is divided by $p_2(x)$ has a non-zero constant term.

SPARSE-POLY-REMAINDER. Given two sparse polynomials $p_1(x)$ and $p_2(x)$ with integer coefficients, the problem is to determine the degree of the remainder when $p_1(x)$ is divided by $p_2(x)$.

Theorem 6.1. SPARSE-POLY-QUOTIENT *and* SPARSE-POLY-REMAINDER *are both* NP-*hard.*

Proof. Consider the function $(1-x^{-a_1}-x^{-a_2}-\cdots-x^{-a_m})^{-1}$. The coefficient of x^{-n} in the power series expansion of this function is non-zero (and positive) iff n can be expressed as a sum using the integers $a_1, a_2, \ldots, a_m$ zero or more times. Therefore, the constant term in the power series expansion of $x^n/(1-x^{-a_1}-x^{-a_2}-\cdots-x^{-a_m})$ is non-zero iff n can be expressed as such a sum. Let $q(x)$ be the quotient when x^{n+c} is divided by $x^c-x^{c-a_1}-x^{c-a_2}-\cdots-x^{c-a_m}$, where c is some integer greater than all the a_i. Then the constant term in $q(x)$ is non-zero (and positive) iff n can be expressed as $\sum_{j=1}^{M} b_j a_j$ with all b_j nonnegative. It is not difficult to see that this latter problem is NP-complete by reducing from the subset sum problem [5]. Hence, taking $p_1(x)$ to be x^{n+c} and $p_2(x)$ to be $x^c-x^{c-a_1}-\cdots-x^{c-a_m}$, we obtain that SPARSE-POLY-QUOTIENT is NP-hard.

For SPARSE-POLY-REMAINDER, we let $p_1(x)$ be x^{n+c-1} and let $p_2(x)$ be as before. Then the degree of the remainder will be $c-1$ iff n can be expressed as above, and will be less than $c-1$ otherwise. Thus SPARSE-POLY-REMAINDER is also NP-hard. This result may have implications for polynomial interpolation, since polynomial interpolation can be viewed as taking the remainder when one polynomial is divided by another. □

Note that SPARSE-POLY-REMAINDER gives a natural *function* which is NP-hard in the sense defined in Section 2, namely, the degree of the remainder. Also, SPARSE-POLY-QUOTIENT gives a natural function which is CoNP-hard in the sense defined in Section 2, namely, the value of the constant term in the remainder.

7. Miscellaneous problems

We now give some NP-hard problems involving the divisibility of sparse polynomials having coefficients of 0 and 1. Also, we obtain results about the divisibility of sparse binary numbers. Finally, we discuss a result about ring homomorphisms.

Consider the following problems.

0,1-SPARSE-POLY-NONFACTOR. Given sets $R1$ and $R2$ of sparse polynomials with coefficients 0 or 1, the problem is to determine whether $\Pi R1$ is not a factor of $\Pi R2$.

SPARSE-BINARY-DIVIS. Given $R1$ and $R2$ as above, the problem is to determine whether $\Pi\{p(2): p \in R1\}$ does not divide $\Pi\{p(2): P \in R2\}$.

Theorem 7.1. 0,1-SPARSE-POLY-NONFACTOR *and* SPARSE-BINARY-DIVIS *are* NP-*hard.*

Proof. We show that if C is a clause, then $\text{Poly}_M(C)$ and $\text{Poly}_M(\neg C)$ can be expressed as $r(x)(x-1)$ or as $r(x)$, where $r(x)$ is a rational function of sparse polynomials whose non-zero coefficients are all $+1$. Furthermore, if C has 3 or less literals, this can be done in polynomial time. We assume that M is a product of small primes and that the prime factorization of M is given. First, observe that $x^b - 1$ can be so expressed, if b is a product of small primes. It is only necessary to apply the identity

$$x^{b_1 b_2} - 1 = (1 + x^{b_1} + x^{2b_1} + \cdots + x^{(b_2-1)b_1})(x^{b_1} - 1)$$

repeatedly, choosing b_2 to be a small prime at each step. Since $\text{Poly}_M(C)$ and $\text{Poly}_M(\neg C)$ can be expressed as rational functions of polynomials of the form $x^b - 1$, we obtain that they can be expressed as $r(x)(x-1)^j$ for some integer j. Note that $r(1)$ is rational and positive. If j were negative, there would be a singularity at $x = 1$, which is impossible. If $j > 1$, then the multiplicity of the zero at $x = 1$ is greater than one, which is also impossible. Hence $j = 0$ or 1 and the desired result follows. If C has 3 or less literals, the expression for $\text{Poly}_M(C)$ and $\text{Poly}_M(\neg C)$ in terms of polynomials of form $x^b - 1$ can be computed in polynomial time, so the expression as $r(x)$ or $r(x)(x-1)$ can also be obtained in polynomial time.

Now, a set S of clauses is inconsistent iff $x^M - 1$ divides $\prod\{\text{Poly}_M(\neg C): C \in S\}$. This latter product can be expressed as $r_1(x)r_2(x)\cdots r_k(x)(x-1)^j$ for some j, where the $r_i(x)$ are rational functions as above. If $j = 0$, then S is consistent. If $j > 0$, then S is inconsistent iff $(x^M - 1)/(x-1)$ divides $r_1(x)r_2(x)\cdots r_k(x)$. But $(x^M - 1)/(x-1)$ can be expressed as a product of sparse polynomials with 0, 1 coefficients, as before. Thus we obtain that 0,1-SPARSE-POLY-NONFACTOR is NP-hard.

To show that SPARSE-BINARY-DIVIS is NP-hard, we apply the usual methods as given in [12]. In fact, a similar result holds for any integer other than 0 and ± 1. We may view $p(2)$ for such a polynomial p to be a 'sparse' binary number. Thus

these results may be interpreted in terms of divisibility problems for integers in sparse n-ary notation for various n. □

Consider the following problem.

RING HOMOMORPHISM INJECTION. Given sparse polynomials $p(x)$, $p_1(x)$, and $p_2(x)$ such that $p_j(x)$ divides $p(x)$ for $j = 1, 2$. Then the problem is to determine whether the 'natural' mapping from $Z[x]/p(x)$ into $(Z[x]/p_1(x)) \times (Z[x]/p_2(x))$ is 1–1. Here $Z[x]$ is the ring of polynomials in one variable over the integers, and p and p_j are assumed to be polynomials in $Z[x]$. The elements of this Cartesian product are ordered pairs of elements from the rings, and operations are done componentwise. The 'natural' mapping is the one that maps a polynomial $q(x)$ in $Z[x]/p(x)$ onto $\langle q(x) \bmod p_1(x), q(x) \bmod p_2(x) \rangle$.

Theorem 7.2. RING HOMOMORPHISM INJECTION *is* NP-*hard.*

Proof. This mapping is not 1–1 iff there is some polynomial $q(x)$ of degree greater than zero, such that $q(x)$ maps onto zero. Such a polynomial $q(x)$ must be a multiple of $p_1(x)$ and $p_2(x)$, hence must be a multiple of $\text{lcm}(p_1, p_2)$. Let $p(x)$ be $p_1(x)p_2(x)$. Then such a q exists iff $p(x)$ is not a factor of $\text{lcm}(p_1, p_2)$, that is, iff p_1 and p_2 are not relatively prime. However, by the problem 2-SPARSE-POLY-GCD it is NP-hard to determine whether two sparse polynomials are not relatively prime. Hence RING HOMOMORPHISM INJECTION is NP-hard. Note that RING HOMOMORPHISM INJECTION may not be in NP since such a q could have many non-zero coefficients compared to p_1 and p_2.

Acknowledgment

The author would like to thank Leonard Adleman who brought Linnik's theorem to his attention.

References

[1] L. Adleman and K. Manders, Reducibility, randomness, and intractability, *9th Ann. ACM Symp. on Theory of Computing* (1977) pp. 151–163.

[2] L. Adleman and K. Manders, Reductions that lie, *Proc. 20th Annual Symp. on Foundations of Computer Science* (1979) pp. 397–410.

[3] G. Birkhoff and S. MacLane, *A Survey of Modern Algebra* (MacMillan, New York, 3rd ed., 1965) pp. 387–389.

[4] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, San Francisco, 1979).

[5] R. Karp, Reducibility among combinatorial problems, in: R. Miller and J. Thatcher, eds., *Complexity of Computer Computation* (Plenum Press, New York, 1972) pp. 85–104.

[6] J.C. Lagarias and A.M. Odlyzko, Effective versions of the Chebotarev Density Theorem, in: A. Frohlich, ed., *Algebraic Number Fields, L-functions, and Galois Properties* (Academic Press, New York, 1977) pp. 409–464.
[7] J.C. Lagarias, Succinct certificates for the solvability of binary quadratic Diophantine equations, *Proc. 20th Ann. Symp. on Foundations of Computer Science* (1979) pp. 47–54.
[8] S. Lang, *Algebra* (Addison-Wesley, Reading, MA, 1970) p. 206.
[9] W.J. Leveque, *Topics in Number Theory Vol. II* (Addison-Wesley, Reading, MA, 1956) pp. 201–228.
[10] U.V. Linnik, On the least prime in an arithmetic progression, I, II, *Mat. Sb. (N.S.)* **15** (56) (1944) 137–178; **15** (57) (1944) 347–368.
[11] K. Manders and L. Adleman, NP-complete decision problems for quadratic polynomials, *Proc. 8th Ann. ACM Symp. on Theory of Computing* (1976) pp. 23–29.
[12] D. Plaisted, Some polynomial and integer divisibility problems are NP-hard, *SIAM J. Comput.* **7** (1978) 458–464.
[13] D. Plaisted, Sparse complex polynomials and polynomial reducibility, *J. Comput. System Sci.* **14** (1977) 210–221.
[14] V. Pratt, Every prime has a succinct certificate, *SIAM J. Comput.* **4** (1975) 214–220.